Université de Montpellier
Faculté des Sciences

MASTER 2 INFORMATIQUE
Parcours Informatique Pour les Sciences

RAPPORT DE STAGE

Intégration et visualisation de données issues du projet Patrimoine Numérique Scientifique du Cirad

Effectué au Cirad du 1er février au 28 juillet 2017 par Thierry Bonnabaud La Bruyère.

Directeur de stage en entreprise : Sandrine Auzoux (Cirad-UPR Aïda)
Co-encadrants : Sophie Fortuno, Mathieu Roche (Cirad-UMR Tetis)

Directeur de stage de l'Université : Gregory Lafitte

2016/2017

Résumé

Le sujet du stage s'inscrit dans le cadre du projet Patrimoine Numérique Scientifique du Cirad qui vise à gérer, conserver et valoriser les données scientifiques ou données de la recherche produites par l'établissement et ses partenaires. Ici nous nous intéressons aux métadonnées décrivant ces ressources. Le travail consiste à la mise en œuvre d'un nouvel inventaire de ces métadonnées par leur normalisation et leur intégration dans une nouvelle base de données. L'inventaire est une application web devant remplacer l'ancien Datacatalog, en offrant un accès aisé aux ressources (lecture, édition) et en permettant de visualiser par des graphiques adaptés l'ensemble de la production scientifique du Cirad selon divers aspects.

Abstract

The subject of the work placement falls under the "Patrimoine Numérique Scientifique" project of Cirad which aims to manage, preserve and value the scientific data or data from the institute and its partners. Here we care about metadata describing these resources. The work is to implement a new inventory of these metadata with their normalization and their integration in a new database. The inventory is a web application wich aims to replace the old Datacatalog and to visualize by appropriate chart the whole scientific production under different aspects.

Mots-clés

data visualisation ; datamart ; web application ; metadata ; patrimoine numérique scientifique ;

Remerciements

Je souhaite remercier ma tutrice Sandrine Auzoux ainsi que mes encadrants Sophie Fortuno et Mathieu Roche pour l'accueil réservé et le temps consacré lors de mon stage de 6 mois au sein du Cirad. Cette première expérience m'aura permis d'appréhender le domaine en pleine expansion qu'est la visualisation de données ainsi que de consolider mes récentes connaissances dans les technologies de web.

J'ai également eu plaisir à travailler en compagnie de mes camarades Bruno d'Agostino et Yu Du, leur plus grande expérience m'aura été profitable.

Table des matières

Table des figures

Chapitre 1

Introduction

1.1 Le Cirad

Le Cirad est un organisme français de recherche agronomique et de coopération internationale pour le développement durable des régions tropicales et méditerranéennes. C'est un Établissement public à caractère industriel et commercial placé sous la double tutelle du ministère de l'Éducation nationale, de l'Enseignement supérieur et de la Recherche et du ministère des Affaires étrangères et du Développement international.

Ses activités relèvent des sciences du vivant, des sciences sociales et des sciences de l'ingénieur appliquées à l'agriculture, à l'alimentation, à l'environnement et à la gestion des territoires. Il travaille autour de 6 grands axes thématiques centrés sur la sécurité alimentaire, le changement climatique, la gestion des ressources naturelles, la réduction des inégalités et la lutte contre la pauvreté [1].

Le Cirad est composé de quatre départements :
- **Dg :** la direction générale.
- **Bios :** le département scientifique Systèmes biologiques mène des recherches sur le vivant, sa caractérisation et son exploitation.
- **Es :** le département scientifique Environnements et sociétés centre ses recherches sur les relations entre agriculture, gestion des ressources naturelles et dynamiques sociales, en lien avec les politiques publiques.
- **Persyst :** le département scientifique Performances des systèmes de production et de transformation tropicaux conduit des études sur les productions tropicales (agriculture familiale et production de rente) à l'échelle de la parcelle, de l'exploitation et de la petite entreprise de transformation. Ses travaux sont réalisés en partenariat avec les acteurs locaux de la recherche en Afrique, Asie, Amérique latine et dans les départements français d'outre-mer.

1.2 Le projet Patrimoine Numérique Scientifique

Le projet Patrimoine Numérique Scientifique (PNS) du Cirad est un chantier d'Établissement lancé en 2013, qui vise à gérer, conserver et valoriser les données scientifiques ou données de la recherche produites par le Cirad et ses partenaires. Dans ce contexte, de nombreux groupes de travail ont permis de contribuer à l'identification des données et d'experts portant des cas d'étude thématiques très prometteurs.

1. `http://www.cirad.fr/qui-sommes-nous/le-cirad-en-bref`

1.3 Objectif

De manière concrète, les unités de recherche du Cirad, en particulier les unités de recherche AIDA, TETIS et SELMET, se sont fortement mobilisées pour constituer un inventaire précis de leurs données scientifiques. Les jeux de données inventoriés contiennent un certain nombre d'informations (métadonnées au format Dublin Core, norme ISO 15836), par exemple, type de données, pays d'exécution, couverture temporelle, thématiques Cirad, auteurs, etc.

Le travail demandé dans le cadre de ce stage consiste à intégrer et normaliser ces métadonnées issues de l'inventaire dans une nouvelle base de données et de fournir des visualisations adaptées en intégrant le tout dans une application web.

1.4 Déroulement du stage

La première étape du stage a été la conception du schéma de la base de données qui a pris les trois premières semaines de février où celui-ci a été révisé avec de profondes modifications notamment du fait des échanges entre la Direction des Systèmes d'Information (DSI) et les chercheurs. Ces discussions étant de nature institutionnelle elles ne seront pas développées ici. Le schéma a été stabilisé mi-mars, en subissant entre-temps des modifications mineures, sans grande incidence sur les autres parties ayant eu cours à ce moment-là. La deuxième étape fut la création des scripts d'extraction et de normalisation qui a duré près de cinq semaines. Les phases de restructuration de la base de données et de normalisation ont été indispensables afin de pouvoir exploiter et visualiser les données sinon on se retrouvait souvent avec des doublons, le cas des noms des auteurs en est l'exemple le plus important (un même auteur pouvait apparaître plusieurs fois sur le graphique), ou plus simplement avec des valeurs erronées pouvant être dû entre autres à une mauvaise saisie.

Cela a été suivi dans un troisième temps par quatre semaines pour réaliser la maquette de l'application web (directement en HTML/CSS pour éviter d'avoir à gérer une grande partie de l'aspect graphique plus tard) qui a aussi servi d'environnement de test pour manipuler différentes visualisations de données. Au cours de cette période quelques jours ont été consacrés à la préparation du serveur vierge qui a été mis à disposition par le Cirad pour accueillir la base de données et l'application web. J'ai ensuite développé les pages web listant les entrées, les filtres de recherche et la vue détaillée (permettant de détailler les entrées individuellement), ceci ayant pris une semaine et demi. Cependant le formulaire fut un peu plus rapide à réaliser. L'implantation des visualisations a occupé le reste du temps.

Voir le Gantt en annexe B.1 pour plus de détails.

Chapitre 2

Problématique

2.1 Les données

Les données qui nous intéressent ici décrivent les différentes ressources numériques produites par les chercheurs du Cirad. Elles nous informent entre autres quels sont les auteurs, quelles sont les unités de recherche impliquées, avec quelles filières agronomiques (agrumes, ananas, coton, etc.), thématiques scientifiques (biodiversité, eau et agriculture, etc.), etc. elles sont associées. Ces informations forment nos entrées dans l'inventaire et place la ressource dans son contexte.

Ainsi une ressource numérique, parfois incluse dans un projet, est située dans le temps (date de mise à jour, couverture temporelle) et l'espace (pays d'exécution, coordonnées géographiques). Elle est associée à une ou plusieurs familles de données, filières, thématiques, unités de recherche, etc.

Par exemple la production de ce stage sera en partie présentée ainsi :

```
-------------------------------------------------------------------------------
Nom court : DataPNS.

Nom complet : Intégration et visualisation de données issues du projet
              Patrimoine Numérique Scientifique du Cirad.

Description : Le sujet du stage s'inscrit dans le cadre du projet
              Patrimoine Numérique Scientifique du Cirad qui vise à gérer,
              conserver et valoriser les données scientifiques ou données
              de la recherche produites par l'établissement et ses partenaires.
              Ici nous nous intéressons aux métadonnées décrivant ces ressources.
              Le travail consiste à la mise en œuvre d'un nouvel inventaire de
              ces métadonnées par leur normalisation et leur intégration dans
              une nouvelle base de données. L'inventaire est une application web
              devant remplacer l'ancien Datacatalog, en offrant un accès aisé
              aux ressources (lecture, édition) et en permettant de visualiser
              par des graphiques adaptés l'ensemble de la production scientifique
              du Cirad selon divers aspects.

Filières agronomiques : Aucune.
```

```
Thématiques scientifiques : Aucune.

Types de données : Base de données ; Logiciel ; Service Portail, Web.

Formats des données : Python ; Javascript ; JSON.

Unités de recherche : UPR AIDA (Persyst) ; UMR TETIS (ES).

Etc.
--------------------------------------------------------------------------------
```

La description des ressources numériques est importante afin de pouvoir les classer et surtout faire émerger de nouvelles connaissances qui pourront être mises en valeur par la visualisation des données.

2.2 L'inventaire à l'origine

Jusqu'à maintenant l'inventaire des données reposait sur une base de données selon une structure particulière (composée en étoile et mise à disposition via une vue SQL représentant une seule table). Cette table utilisait des clés "mouvantes" pour identifier les entrées de l'inventaire, c'est-à-dire qu'elles ne disposaient pas de clés primaires. Cela est dû au fait que la base de données prenait sa source dans des documents Excel (.xlsx) et qu'elle était vidée toute les nuits pour s'actualiser grâce à un script batch qui prenait en compte les documents Excel nouvellement créés ou mis à jour. De ce fait, les identifiants n'étaient pas les mêmes d'une mise-à-jour à l'autre. Il n'y avait donc pas de réelle persistance des données.

Les documents Excel (voir un exemple à l'annexe C.1) sont à ce jour le seul moyen pour entrer les informations et la tâche est fastidieuse pour les chercheurs, notamment parce qu'ils contiennent plusieurs onglets et plus de 50 colonnes. En ce qui concerne les listes des termes CIRAD, il n'y a pas de saisie automatique. Les chercheurs doivent aller sur d'autres onglets spécifiques chercher le terme souhaité et le copier dans le colonne correspondante dans la fiche principale. Ce qui occasionne des erreurs de saisie et rend complexe l'exploitation des données. De plus, il n'y a pas de contraintes d'intégrité (domaines référentiels, etc.).

L'inventaire contenait à ce jour 370 entrées (dont 348 visibles, c'est-à-dire dont l'affichage aux autres usagers est autorisé et n'étant pas sujet à la confidentialité) associées à 245 projets et comprenant plus de 850 auteurs ou contributeurs différents.

Pour résoudre les problèmes cités précédemment et pour avoir catalogue des données pérenne et de qualité il s'est avéré opportun d'élaborer un nouveau schéma de base de données relationnelle adapté à un annuaire de données de "production". Celui-ci devrait être relié à terme à un entrepôt de données (data warehouse [1]), tel que Dataverse [2], qui l'alimentera (en plus des données existantes).

Cette base de données relationnelle est un datamart, c'est-à-dire un sous-ensemble d'un data warehouse destiné à être interrogé sur un panel de données restreint à son domaine fonctionnel, selon des paramètres qui auraient été définis à l'avance lors de sa conception. C'est une sorte de vue sur un entrepôt de données.

1. Un data warehouse est une base de données utilisée pour collecter, ordonner, journaliser et stocker des informations provenant de base de données opérationnelles.
2. `http://dataverse.org/about`

2.3 Application Web et visualisation des données

À ce jour un site intranet existe, le "Datacatalog" (voir annexe C.2), pour permettre de lister les entrées de l'inventaire. Il comprend une vue liste et une vue détaillée contenant toutes les informations d'une entrée. Il dispose aussi de filtres de recherche et d'un index géographique pour atteindre les ressources par l'intermédiaire d'une carte ArcGIS [3]. Cependant, il termine son service à la fin de l'année 2017 et de ce fait il faut le remplacer par une application plus adaptée facilitant la consultation et la saisie des ressources numériques et proposant plus de fonctionnalités.

Cet ensemble de métadonnées est difficile à analyser dans sa globalité d'un rapide coup d'œil. Par exemple, à ce jour nous ne pouvons pas comparer efficacement la quantité de ressources fournies par unité, contributeur, etc. mais seulement opérer une recherche affinée avec ces paramètres. Nous avons besoin pour cela de représentations graphiques adéquates (diagrammes ou autres) et ergonomiques. Cependant les visualisations ne doivent être ni trop simples (sauf si la situation l'exige) ni trop complexes, c'est-à-dire ne pas représenter trop d'informations à la fois. Il faut ainsi minimiser l'usage des simples diagrammes circulaires ou histogrammes car ils ne donnent qu'une information.

De manière générale, la visualisation de données est un ensemble de méthodes de représentation graphique visant à représenter des ensembles complexes de données, de manière plus simple, didactique et pédagogique et pouvant servir à l'aide à la décision. Les outils de reporting et la business intelligence utilisent ces méthodes mais se limitent souvent aux simples diagrammes mentionnés plus haut. Aujourd'hui ces techniques permettent la création de visualisations uniques et parfaitement adaptées au problème étudié, en ayant parfois la capacité de de faire émerger des détails cachés qui n'auraient pas été remarqués autrement.

2.4 Méthodologie

Le stage suit une ligne intermédiaire entre le stage professionnel et le stage recherche. En particulier la partie visualisation des données qui se veut exploratoire. De ce fait les objectifs ce sont trouvés régulièrement changés et parfois en profondeur. Pour cela nous avons adopté avec les encadrants pour une attitude pragmatique en travaillant de manière purement agile [4]. Celle-ci a donc consisté en de multiples réunions hebdomadaires afin de progresser pas à pas et avec souplesse.

2.5 Outils utilisés

PostgreSQL 9.5 a été choisi comme base de données car c'est un logiciel libre avec une extension PostGIS qui offre un support d'objets géographiques à la base de données. Le modèle physique a été réalisé à l'aide de pgModeler [5]. Les divers scripts pour les phases de normalisation et d'intégration ont été écrits en Python (version 3), notamment pour la clarté du code et la rapidité de mise en œuvre.

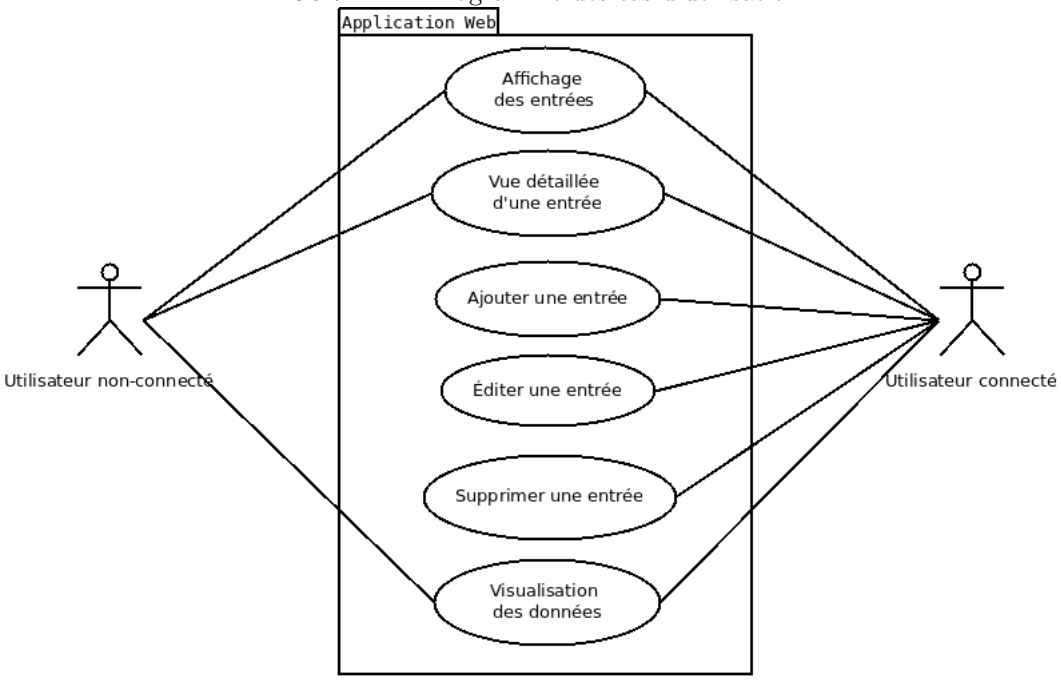
Parmi les bibliothèques Python utilisées nous avons : OpenPyXL pour ouvrir les documents Excel (XSLX) et Psycopg2 pour la jonction avec PostgreSQL.

3. `https://www.esrifrance.fr/arcgis.aspx`
4. `http://agilemanifesto.org/iso/fr/manifesto.html`
5. `https://www.pgmodeler.com.br/`

Le framework Django a été utilisé pour le développement de l'application web pour les mêmes raisons mentionnées précédemment, avec la possibilité de réutiliser une partie des scripts de manière naturelle. Les aspects dynamiques ont été écrits en Javascript (généralement avec jQuery), de même que la partie "visualisation des données" (avec D3.js [6] pour les visualisations les plus complexes et Chart.js [7] pour les plus simples). Le framework CSS bootstrap est utilisé pour améliorer l'expérience utilisateur. L'application est hébergée sur un serveur CentOS.

2.6 Cas d'utilisation

FIGURE 2.1 – Diagramme des cas d'utilisation



6. https://d3js.org/
7. http://www.chartjs.org/

Chapitre 3

Intégration des données

3.1 Conception de la base de données

Nous commençons donc ici par nettoyer les attributs existants pour ne garder que ceux qui seront utiles au nouveau schéma de la base de données.

3.1.1 Nettoyage des attributs

Suite à des erreurs de saisie, des lignes incomplètes (seulement quelques attributs mineurs sont remplis) apparaissent dans les documents Excel. Certains champs/attributs seront modifiés, d'autres seront supprimés car liés au processus d'inventaire et obsolètes désormais. Les voilà plus détaillés ci-dessous :

Modifiés :

- Lieux (villes, villages) : éléments séparés par des points-virgules → Description des localisations : texte libre
- Mesures, données chiffrées : booléen → Nature des données : ['Qualitatif', 'Quantitatif', 'Quantitatif et qualitatif', 'À déterminer']
- Géo-référencement : texte → booléen
- Période d'embargo : booléen → texte (['Oui', 'Non', 'Ne sais pas'] avec les données actuelles)
- Auteurs : texte + Contributeurs : texte + Contacts ressource : texte → Personne : table (prénom, nom, email, affiliation)
- Publication(s) : ['Oui, une publication', 'Oui, plusieurs publications', 'Non'] → Lien vers la publication principale : texte
- Unité : Un-À-Plusieurs → Plusieurs-À-Plusieurs

Supprimés :

- Zone géographique (vaste région, imprécise et source d'erreurs)
- Description localisation
- Score 1 et 2 : [0, 1, 2, 3]
- Présence d'images : booléen
- Présence de documents méthodologiques : booléen
- Annotations par l'unité : texte libre
- Date de mise en annuaire : date
- Ressource biologique : booléen

— Séquence omique : booléen
— Précisions autres, sur le format : texte libre
— Documentation méthodologique : ['pas necessaire', 'oui nécessaire et disponible', 'oui mais plus disponible']

Ajoutés :

— Famille (de données) : [séquence omique, ressource biologique, etc.]
— Commune (plus petite division administrative) : table (nom et position spatial)
— Unité géographique : table (description ,échelle, point et polygone)
— Organisme : table (nom)
— ID persistent (entrepôt de données, etc.) : texte
— Nom de l'entrepôt source : texte
— Statut : texte ('Draft, unpublished', etc.)
— Métadonnées JSON : json (pour un éventuel nouveau groupe de métadonnées)

3.1.2 Nouveau schéma

Le schéma devant correspondre à certains éléments de métadonnées du Dataverse, il a fallu collaborer avec la Direction des Systèmes d'Information afin de se concerter sur la voie à suivre. Ainsi après de longs échanges nous sommes arrivé à notre modèle physique actuel (en annexe C.3 pour des raisons pratiques). Il est à noter que pgModeler utilise son propre formalisme pour représenter les relations, il ne s'agit ni de l'UML, ni du Merise.

Les tables Personne et Organisme ont un traitement à part. Elles disposent chacune d'une relation Plusieurs-À-Plusieurs avec l'entité Entree sous forme de triplet, le troisième attribut étant une clé étrangère pointant vers une table référençant les différents rôles possibles ($role\_personne$ ou $role\_organisme$). La première caractérise le rôle d'une personne tel que auteur, contributeur ou contact. La deuxième indique le rôle d'un organisme tel que déclarant ou partenaire. Les clés étrangères pointant vers les rôles sont des chaînes de caractères pour faciliter les requêtes en évitant des jointures supplémentaires.

3.2 Normalisation des données

Certains champs d'une feuille Excel sont reliés à un référentiel du Cirad contenu dans les autres onglets du document (accessibles depuis une liste déroulante pour la saisie) rendant leur normalisation *a priori* inutile. Une partie des données initiales doit être normalisée car nous pouvons rencontrer des casses différentes, des fautes ou même spécifiquement pour les contributeurs l'inversion nom-prénom/prénom-nom.

Par exemple :

```
G. Dupont
DUPONT Georges
G Dupond
G. Dupont (georges.dupond@cirad.fr)
```

Les données impossibles à traiter automatiquement ont été intégrées manuellement. Ces opérations importantes en terme de temps sont réalisées à l'aide de plusieurs scripts écrits en langage Python et utilisent un jeu d'expressions régulières adéquates.

3.2.1 Dates

Pour des raisons inconnues certaines dates présentaient des erreurs de formatage et se trouvaient ainsi sous la forme d'un entier (tel que 42755, produit parfois par Excel). Elles ont été corrigées par la fonction suivante :

```
1 def goodDate(date):
2     """ if int value like 42755 return date value """
3     if isinstance(date, int) and date > 10000:
4         return from_excel(date, offset=CALENDAR_WINDOWS_1900)
5     else:
6         return date
```

3.2.2 Pays

Bien que les noms de pays utilisés dans les données initiales soient issues d'un référentiel du Cirad il est préférable de choisir des termes standardisés plus usités. Pour cela deux services web ont été retenus pour comparaison : GeoNames [1] (plus complet) et Nominatim [2]. Tous deux sont des bases de données géographiques gratuites référençant des noms (plusieurs langues), des coordonnées, des codes postaux...

Le script Python paysToStandardNames.py établit une correspondance entre les codes ISO3 du référentiel du Cirad et les noms français issus des services GeoNames (un fichier JSON contenant toutes les informations nécessaires est récupéré au préalable) et Nominatim (lequel est accessible uniquement par des requêtes). Il crée ainsi un nouveau fichier JSON qui sera exploité plus tard dans le processus d'intégration.

1. http://www.geonames.org/
2. http://wiki.openstreetmap.org/wiki/Nominatim

Par exemple :

```
{
    "chine": {
        "geonames": "Chine",
        "iso": "CHN",
        "nominatim": "République populaire de Chine"
    },
    "république de moldova": {
        "geonames": "Moldavie",
        "iso": "MDA",
        "nominatim": "Transnistrie"
    }
}
```

Le premier terme est le terme issu du référentiel du Cirad, il contient celui de GeoNames, celui de Nominatim ainsi que le code ISO3 qui nous sert d'identifiant dans la base de données. De part l'exemple ci-dessus il en ressort que GeoNames est plus intéressant car utilise des noms plus courts et plus fiables.

3.2.3 Auteurs, contributeurs et contacts

L'objectif est d'obtenir des noms de personnes propres afin de pouvoir les rentrer correctement dans la table Personne de la base de données, sous la forme {Prénom, Nom, email et affiliation}. Pour cela plusieurs expressions régulières ont été développées au cours du stage.

Les noms étant inscrits dans une zone de texte libre, il s'agit dans un premier temps de séparer les éléments formant une liste. Après analyse, deux types de liste sont apparues.

Le premier type est formé par des noms, e-mails, etc. séparés par des virgules ou des points-virgule. Il est détecté grâce au motif suivant :

```
"([^ ;,]+[^;,]*[^ ;,]+)[;,]?"
```

Le deuxième est formé par des adresses e-mails séparées par des espaces. Il est détecté grâce au motif suivant :

```
"[^ ]+@[^ ]+ +[^ ]+@[^ ]+"
```

Une fois les éléments isolés il reste à extraire le prénom, le nom et l'adresse e-mail si celle-ci est disponible. Parfois seule cette dernière est présente. Il a fallu aussi ignorer certaines valeurs inexploitables qui polluaient l'inventaire, par ce motif :

```
".*[Pp]ersonnel.*|.*[Ss]tagiaire.*|WWF \(Hr\.\)|etc\.|Spot image|Pot D|.*[Pp]rojet.*|
    Université.*|.*[Aa]nnexe.*|& al\.|A compléter|[Cc]ontributeur|.*[Tt]echnicien.*|
    Laboratory Methods|.*Pl@ntNet|DRAAF LR|Département CA|[Cc]onservatoire.*|.*[Hh]
    istorique.*|[Mm]étéo.*|and web.*|ONG AVSF|.*présent.*|CETE sud-ouest|Auteurs
    multiples|unité HORTSYS|Conseil Général de Mayotte"
```

Les personnes sont nommées de plusieurs manières (parfois pour un même auteur) :

- Prénom Nom
- Prénom Nom (autre)
- NOM Prénom
- PRÉNOM NOM
- Nom P
- des expressions contenant "ingénieur en informatique"
- e-mail

Et le contenu de ce champ est capturé par les motifs suivants :

```
# Firstname Name (something)
validNameRegex1 = re.compile("([^ ]+) +(.+) +\(.*")
# NAME Firstname
validNameRegex2 = re.compile("([A-Z ]{2,}) +([A-Z][a-z]+)")
# FIRSTNAME NAME
validNameRegex3 = re.compile("^[A-Z]+\.? +[A-Z]+$")
# Name F
validNameRegex4 = re.compile("(.+[^ ]+) ([A-Z])$")
# Stagiaires d'école d'ingénieur en informatique
stagiairesRegex = re.compile(".*d'ingénieur en informatique : (.+)")
# ident@domain.com
mailRegex = re.compile("^[^ ]+@[^ ]+[^ ,;]*")
```

Parfois certains noms étaient très proches, à une majuscule ou un accent près. La fonction de similarité suivante (calcul de la distance entre deux chaînes de caractères à l'aide la bibliothèque difflib de Python) a permis de les confondre :

```
def similar(seq1, seq2, ratio=0.8):
    """ Return True if the first string is similar to the second """
    seq1 = re.sub("[éè]", "e", seq1.lower()).replace("ï", "i")
    seq2 = re.sub("[éè]", "e", seq2.lower()).replace("ï", "i")

    return difflib.SequenceMatcher(a=seq1, b=seq2).ratio() > ratio
```

Le reste des expressions a été traité manuellement, soit environ une cinquantaine de cas à corriger au fur et à mesure de leur découverte à l'aide de l'annuaire ou simplement en corrigeant les fautes d'orthographe. La tâche a été compliquée par le fait que certains auteurs ne font pas partie du Cirad et de ce fait ne sont pas référencés dans l'annuaire de l'établissement.

3.3 Première insertion des données

Sur mon poste de développement existe un fichier Excel par auteur participant à l'inventaire. Ces documents sont situés dans les répertoires désignés par le nom de l'unité de recherche, eux-même étant répartis dans les répertoires associés aux départements. Pour optimiser la vitesse des essais d'insertion (les tables de la nouvelle base de données étant vidées à chaque fois) un script fusionne l'ensemble des documents en un unique (nommé unique.xlsx), ainsi on évite une succession inutile d'ouverture/fermeture. Le script fusionne aussi les deux premières feuilles "Conventions Unité" et "Lot|WP|Autres ressources", la distinction n'ayant pas lieu d'être dans notre situation. La base de données sera remplie à partir de ce nouveau fichier. Pour cela deux scripts principaux, aidés de scripts intermédiaires, seront créés.

Le premier script permet d'alimenter les tables devant contenir les valeurs communes à tous les documents (la liste des unités, la liste des formats, etc.), les données étant issues du référentiel du Cirad. Ces données étant en partie contenues dans chaque document Excel, dans des feuilles annexes. Le document servant de modèle pour les chercheurs a été pris comme source. La hiérarchie département/unités a été placée manuellement dans un dictionnaire pour des raisons pratiques.

Fonction intermédiaire d'ajout de valeur dans une table :

```
def insertValue(value, tableName):
    """ Insert a value in a given table """
    cur.execute(" INSERT INTO " + tableName + " VALUES (DEFAULT, %s)", (value,))
```

Le deuxième script parcourt chaque ligne du fichier unique et ignore les champs qui n'ont plus de correspondance avec le nouveau schéma relationnel. Si les champs obligatoires d'une ligne sont incomplets celle-ci est alors ignorée. Il affiche à la sortie du terminal les éléments qui n'ont pas pu être insérés.

Ainsi on insère d'abord les projets, les personnes (auteurs, etc.) puis les entrées. À ce moment-là on affecte à diverses variables les champs simples ainsi que les clés étrangères des relations Un-À-Plusieurs, lesquelles sont récupérées par une requête idoine telle que celle-ci :

```
cur.execute(""" SELECT id FROM public."DispositifPartenariat" WHERE nom=%s """,
            (row[49].value,))
id_DispositifPartenariat = cur.fetchone()
```

Tout ceci nous permet de générer une large requête d'insertion :

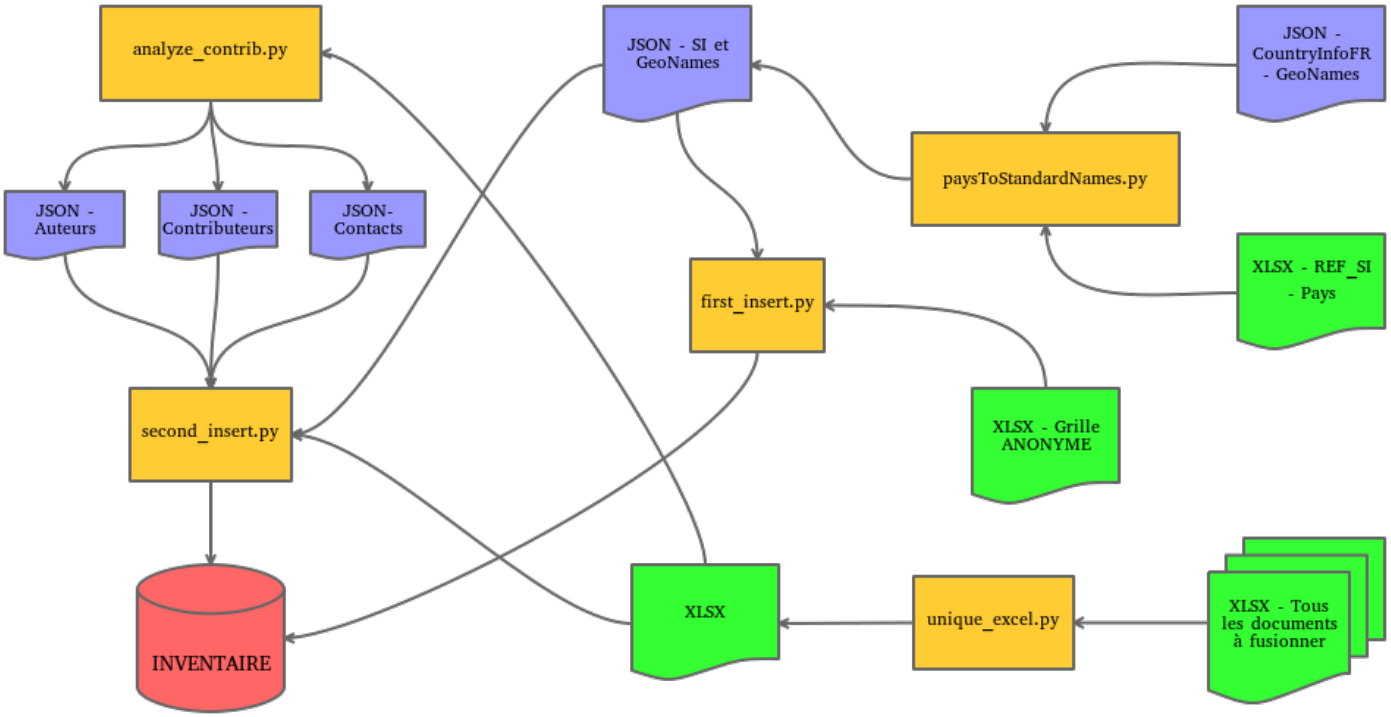
```
 cur.execute(""" INSERT INTO public."Entree"
             VALUES (DEFAULT, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,
             %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, NULL, NULL,
              'Draft, unpublished', NULL) RETURNING id """,
             (dateMAJ, nomCourt, nomComplet, description, motsCles, couvertureTemp,
             natureDonnees, formatPrecisionsAutres, geoRef, lienRessource,
             periodeEmbargo, id_InfraDeRech, autreIR, descOrganPartenaires,
             lienPublication, remarquesGenerales, affResAnnInterne, volumetrie,
             modalitesAcces, niveauPartage, id_DispositifPartenariat,
             id_SupportStockage ))

id_Entree = cur.fetchone()
```

PostgreSQL nous permet à partir de la version 9.5 de retourner la clé primaire (ou autre) du dernier n-uplet inséré grâce au mot-clé RETURNING. Nous pouvons ainsi la stocker et éviter une nouvelle requête pour chercher l'entrée afin de créer les liens de chaque relation Plusieurs-À-Plusieurs. Ces liens sont réalisés à l'aide de fonctions de cette forme :

```
def manyToManyProjet(id_Entree, row):
    """ Fill public."many_Projet_has_many_Entree" table """
    cur.execute(""" SELECT id FROM public."Projet" WHERE nom=%s """,
                (row[2].value,))
    id_Projet = cur.fetchone()

    if id_Projet is not None:
        cur.execute(""" INSERT INTO public."many_Projet_has_many_Entree"
            VALUES (%s, %s) """, (id_Projet, id_Entree))
```

3.4 Schéma des actions de l'ensemble des scripts

FIGURE 3.1 – Schéma des actions de l'ensemble des scripts



En jaune sont représentés les scripts Python, en bleu les fichiers JSON et en vert les documents Excel.

L'ensemble des scripts reposent sur des fichiers JSON (dont le format s'intègre naturellement avec le système de dictionnaire en Python) et Excel en entrée comme en sortie. Sur le schéma ci-dessus nous pouvons voir que les sorties de certains scripts alimentent d'autres scripts avant de remplir la nouvelle base de données.

Chapitre 4

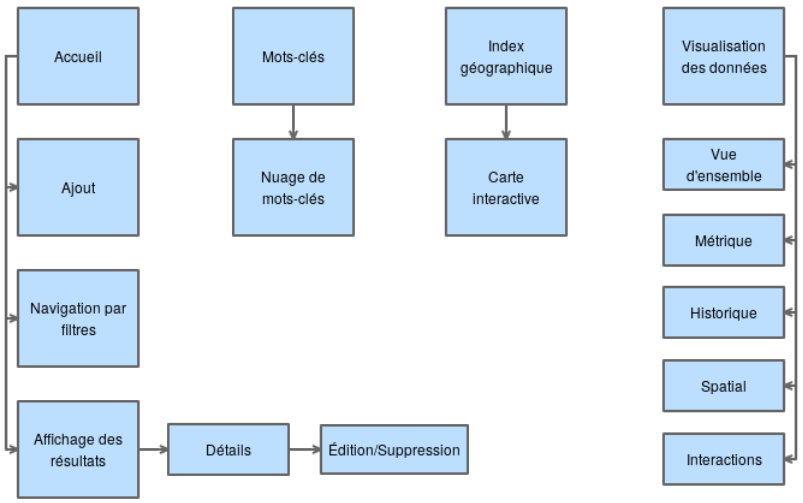
Application Web et visualisation des données

Maintenant que nous avons une nouvelle base de données fraîchement remplie, il nous faut une interface de présentation des données de l'inventaire offrant des visualisations qui permettent d'appréhender de manière graphique l'ensemble des métadonnées.

4.1 Structure de l'application

L'application contient quatre parties : l'accueil, une vue sur les mots-clés, un index géographique et l'ensemble des visualisations des données. La vue sur les mots-clés est censée montrer un nuage de mots-clés interactif et l'index géographique doit permettre d'accéder aux ressources par une carte (cette fois-ci par l'usage de technologies libres telles que leaflet.js et OpenStreetMap). Cependant il a été décidé avec mes encadrants d'en reporter leur étude et leur développement afin de se concentrer sur les deux autres.

FIGURE 4.1 – Schéma de la structure du site



4.2 Vue en liste et filtres de recherche

4.2.1 Liste des résultats

L'affichage des résultats comporte la gestion de la pagination (15 entrées par page). Les entrées de l'inventaire sont affichées à la suite sous forme de pavé d'information. Chaque pavé contient un titre cliquable (pointant vers la vue détaillée), la date de mise-à-jour ainsi qu'une partie des informations de l'entrée :

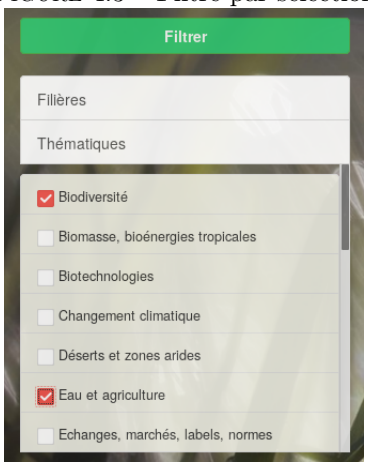
— le résumé
— la couverture temporelle
— les pays d'exécution
— les familles de données (séries temporelles, observations de terrains, etc.)
— les formats des données (XLS, Python, etc.)
— le lien vers la ressource

L'affichage se veut compact pour pouvoir lire facilement et survoler rapidement les résultats. Les éléments d'un champ pouvant en contenir plusieurs sont affichés sous forme de petites étiquettes colorées afin d'avoir une meilleure visibilité.

FIGURE 4.2 – Une partie de l'affichage des résultats

4.2.2 Filtres

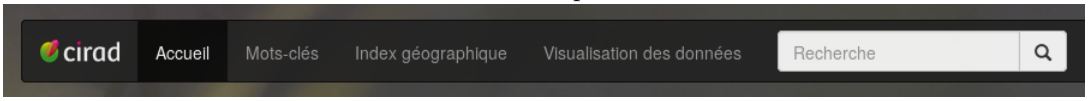
FIGURE 4.3 – Filtre par sélection



Ce filtre est une liste de menus déroulants. Chacun de ces menus est une liste de cases à cocher (pour filtrer selon les relations Plusieurs-À-Plusieurs : filières, thématiques, etc.) ou de boutons radio (pour les relations Un-À-Plusieurs).

Il est aussi possible de faire une simple recherche textuelle. Celle-ci parcourt les champs suivants :
- nom complet
- nom court
- description
- mots-clés
- auteurs, contributeurs et contacts
- description des unités géographiques

FIGURE 4.4 – Simple recherche



4.3 Vue détaillée d'une entrée

La vue détaillée de l'ancien Datacatalog était un simple et gros tableau qui condensait toutes les informations sur une entrée, d'un seul bloc. La vue a donc été rendue plus ergonomique séparant dans plusieurs panneaux les différents éléments pour les grouper selon des considérations administratives (voir la maquette 4.5 et l'annexe C.4). Elle comporte aussi deux boutons : un d'édition et un autre de suppression.

Cette vue comporte sept panneaux. Le premier contient des informations générales telles que la date de mise-à-jour et la description. Elle contient aussi, dans la mesure où elle est présente, les remarques générales.

Le panneau "Affiliations structurelles" contient la liste des projets associés, des unités de recherche (avec le nom du département), des organismes déclarants et partenaires.

Le panneau "Caractérisation scientifique" contient les familles de données, les thématiques, les filières et les mots-clés. Il décrit la nature des données (qualitatif et/ou quantitatif).

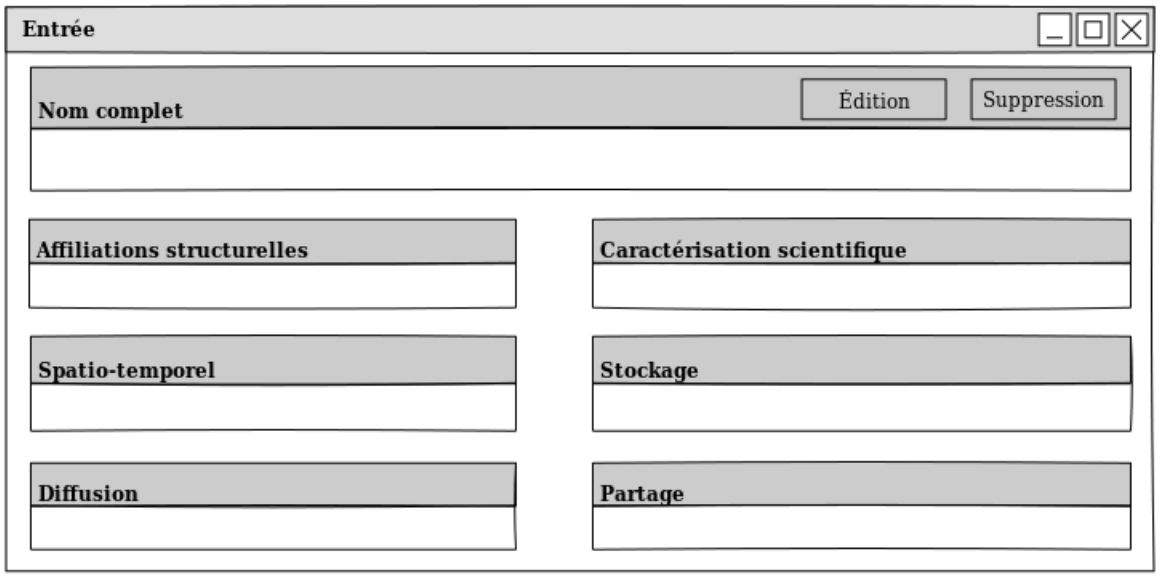
Le panneau "Spatio-temporel" contient la couverture temporelle, les pays d'exécution, les communes (en tant qu'entités administratives) et les unités géographiques en comprenant leur description et leur position.

Le panneau "Stockage" nous indique sur le type des données (logiciel, jeu de données, etc.), leurs formats (XLS, Python), leur volumétrie ainsi que le support de stockage principal.

Le panneau "Diffusion" nous informe sur les différentes personnes ayant travaillé sur la ressource, en séparant les auteurs, les contributeurs et les contacts (ce dernier champ affiche si possible l'adresse e-mail de chaque personne). Nous y trouvons aussi le lien vers la ressource, les langues employées et le lien vers la publication principale associée.

Le panneau "Partage" se contente quant à lui de montrer les modalités d'accès (libre, privé pas de partage, choix du partenaire, etc.), le niveau de partage (en interne tout Cirad, UMR, pas partagé, etc.) et s'il existe une période d'embargo sur la diffusion de la donnée.

FIGURE 4.5 – Maquette de la vue détaillée



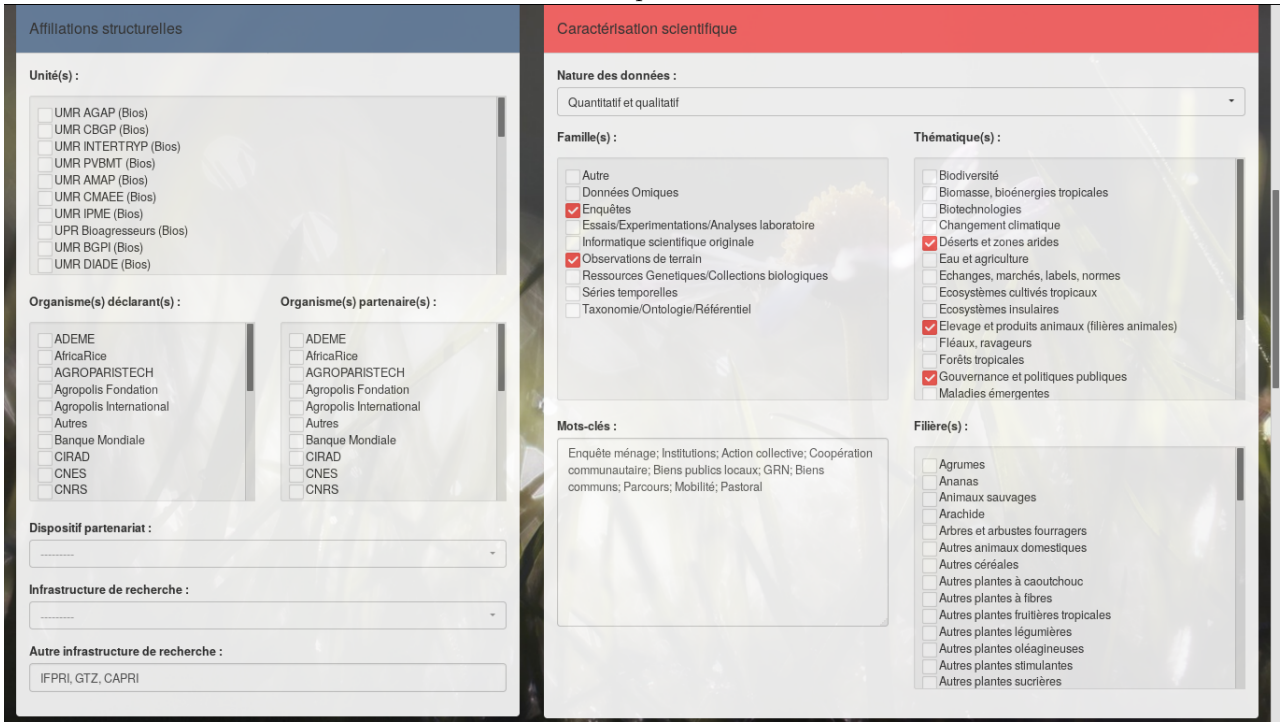
4.4 Formulaire

Le formulaire va remplacer la saisie des informations par l'intermédiaire des documents Excel et ainsi simplifier considérablement l'opération tout en garantissant la normalisation des données essentielles, ce qui sera utile pour la tâche suivante de visualisation. Il dispose des mêmes panneaux que la vue détaillée pour suivre la même structure. Ils disposent de zones de saisie de texte, de listes de cases à cocher ainsi que de sélecteurs avec menu déroulant. Certaines zones de saisie de texte comportent un système de complétion automatique afin de faciliter la saisie de projets et d'auteurs. En effet ces derniers étant nombreux, on ne peut se permettre de les afficher dans une liste de cases à cocher. Une zone textuelle s'est donc avérée plus simple d'usage. La difficulté étant de faire correspondre les noms et les prénoms avec ceux dans la base. Le complétion automatique intervient justement pour pallier ce problème en forçant le formatage suivant :

```
NOM, Prénom ;
```

L'ajout de personne et de projet ne sont pas encore implémentés (travaux de connexions à faire avec le système d'information du Cirad) mais sont supposés être à terme des boutons situés aux endroits adéquats pour faire surgir un petite fenêtre qui permettra une insertion rapide et dynamique dans la base.

FIGURE 4.6 – Une partie du formulaire



4.5 Visualisation des données

Maintenant qu'il est possible d'avoir de nouvelles données normalisées il s'agit de pouvoir les visualiser dans leur ensemble.

La visualisation des données est divisée en cinq parties (vue d'ensemble, historique, interactions, spatial et métrique), mais seules les trois premières ont été développées pour le moment. La partie métrique propose des diagrammes circulaires et la partie spatial devrait contenir une carte interactive (leaflet.js) en proposant diverses colorations selon la densité des données produites par pays, thématiques, etc. Les visualisations suivantes ont pour la plupart été réalisées à l'aide de la bibliothèque javascript D3.js.

4.5.1 Vue d'ensemble

La vue d'ensemble montre la proportion de données signalées en terme d'entrées par rapport au département et aux unités de recherche. Il s'agit d'observer les proportions des éléments de plusieurs tables, notamment grâce à un sélecteur. Pour cette représentation le treemap circulaire (voir figure 4.7) a été retenu. Là où de simples histogrammes ou bien des diagrammes circulaires ne nous auraient permis de voir uniquement un proportion de tel ou tel élément, le treemap circulaire permet d'établir des proportions dans un contexte donné. Par exemple, le diagramme circulaire pourra au mieux exprimer la proportion des entrées pour une seule famille de données. Le treemap pourra quant à lui montrer ces mêmes proportions mais pour toutes les familles en même temps.

Ci-dessous on remarque que les observations de terrains représentent la famille de données majoritaire parmi toutes les entrées et que c'est le département Persyst qui en a inventorié le plus.

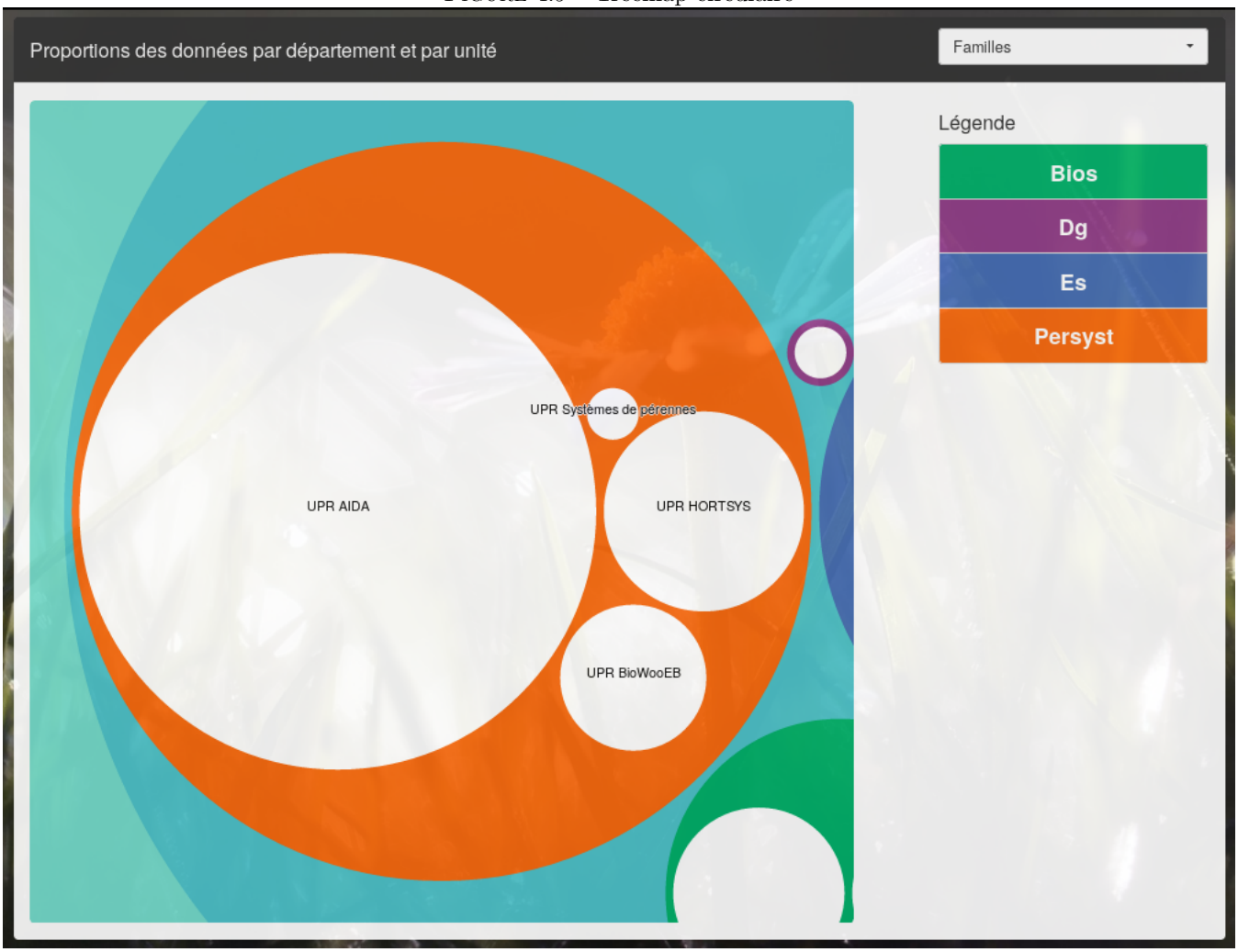
FIGURE 4.7 – Treemap circulaire

Le treemap construit est interactif et par un clic permet de descendre dans l'arborescence, c'est-à-dire zoomer sur les départements (les cercles colorés) et enfin sur les unités de recherche (les cercles blancs, les noms apparaissant après le zoom). Pour rendre la lecture plus efficace il comporte un code couleur qui permet d'identifier rapidement les départements scientifiques au sein de la représentation.

FIGURE 4.8 – Treemap circulaire

Ici nous avons un zoom sur la famille de données "Observations de terrain" qui fait apparaître le nom des départements scientifiques et les proportions des entrées associées.

FIGURE 4.9 – Treemap circulaire

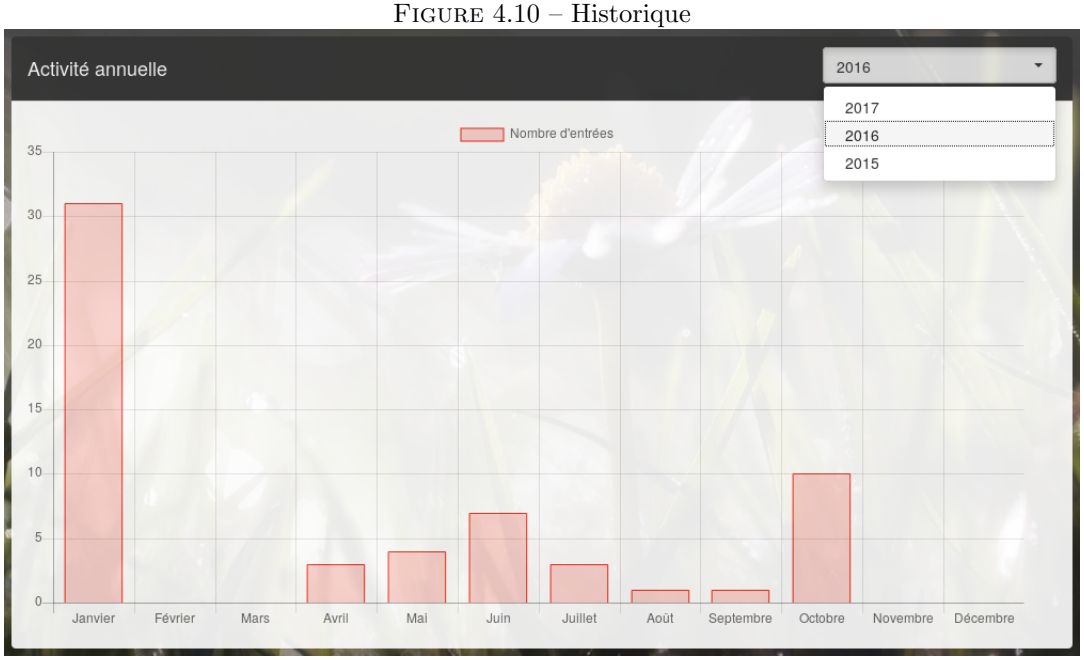


Ici nous avons un zoom sur le département Persyst qui fait apparaître les noms des unités de recherche et les proportions des entrées associées.

4.5.2 Historique

Pour la partie temporelle seule l'activité annuelle est représentée à ce stade. Pour celle-ci un histogramme a été choisi. Il se base sur les dates de mise-à-jour et propose un sélecteur pour choisir l'année d'activité qui nous intéresse. Le changement d'année provoque une animation faisant varier la hauteur des colonnes. Le diagramme a été réalisé à l'aide de la bibliothèque javascript Chart.js.

FIGURE 4.10 – Historique



4.5.3 Interactions

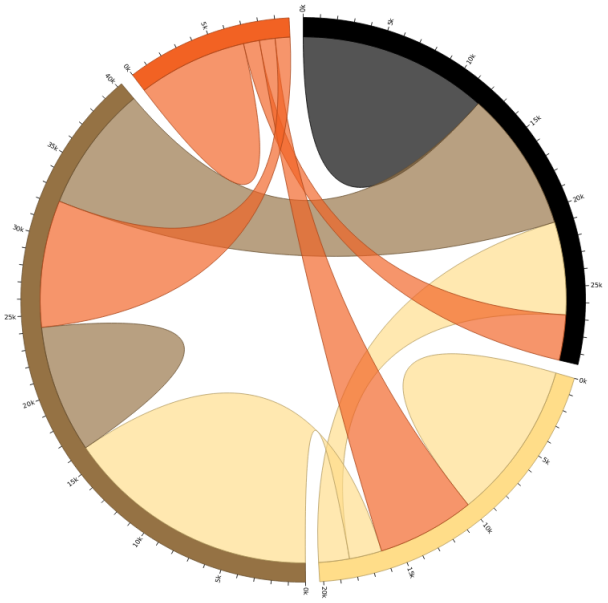
Nous en venons ici à traiter de visualisations plus complexes. Elles sont nommées chord diagram et servent à montrer les interactions entre divers éléments. Elles sont assez prisées en génomique pour établir des correspondances entre génomes ou dans les domaines proches de l'économie pour mettre en évidence des transactions (comme les acquisitions de terres entre pays par exemple [1]).

D3.js étant une boîte à outils pour la visualisation (écrite en Javascript), il n'existe pas de chord diagram prêt à l'emploi. Cependant je me suis basé sur un code existant (l'exemple ci-dessous) pour construire les diagrammes recherchés. J'ai donc dû prendre le temps de l'étudier pour y parvenir, le manque de clarté propre au langage utilisé n'aidant pas.

1. `http://landmatrix.org/en/get-the-idea/web-transnational-deals/`

Un chord diagram est réalisé à l'aide d'une matrice carrée représentant un graphe dirigé. Par exemple le diagramme ci-dessous (figure 4.11) :

FIGURE 4.11 – Exemple de chord diagram



La matrice associée est la suivante :

$$\begin{pmatrix} 11975 & 5871 & 8916 & 2868 \\ 1951 & 10048 & 2060 & 6171 \\ 8010 & 16145 & 8090 & 8045 \\ 1013 & 990 & 940 & 6907 \end{pmatrix}$$

L'élément $x_{i,j}$ donne la largeur du ruban au départ du lien de i vers j. De même l'élément $x_{j,i}$ donne la largeur du ruban à l'arrivée de ce même lien.

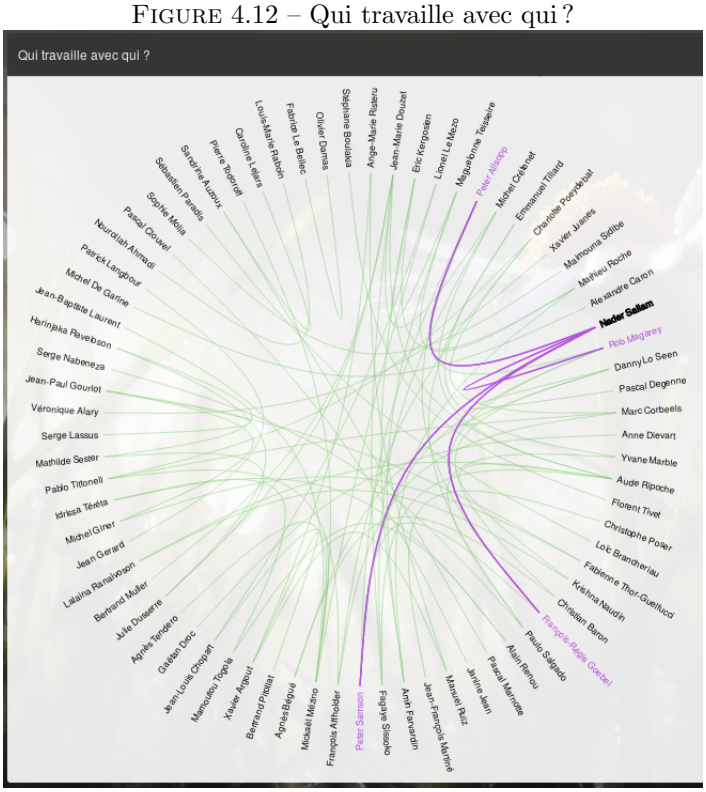
Relation Personnes vers Personnes

Ici on ne représente (voir figure 4.12) que l'existence d'une relation entre telle et telle personne, on ne montre pas la densité de la production personnelle de données scientifiques. Notamment parce que ce n'est pas une information pertinente dans la mesure où un chercheur peut être en poste depuis vingt ans comme un an, ce qui n'est pas comparable.

La base référençant plus de 850 personnes il a été décidé de limiter la visualisation aux personnes ayant au moins trois contributions à leur actif afin d'avoir un résultat plus lisible.

La représentation est interactive. En effet si le curseur est placé au-dessus d'un nom celui-ci devient gras et les liens aux autres personnes passent du vert au violet tout en s'épaississant légèrement. Cette action colore aussi les noms ciblés en violet.

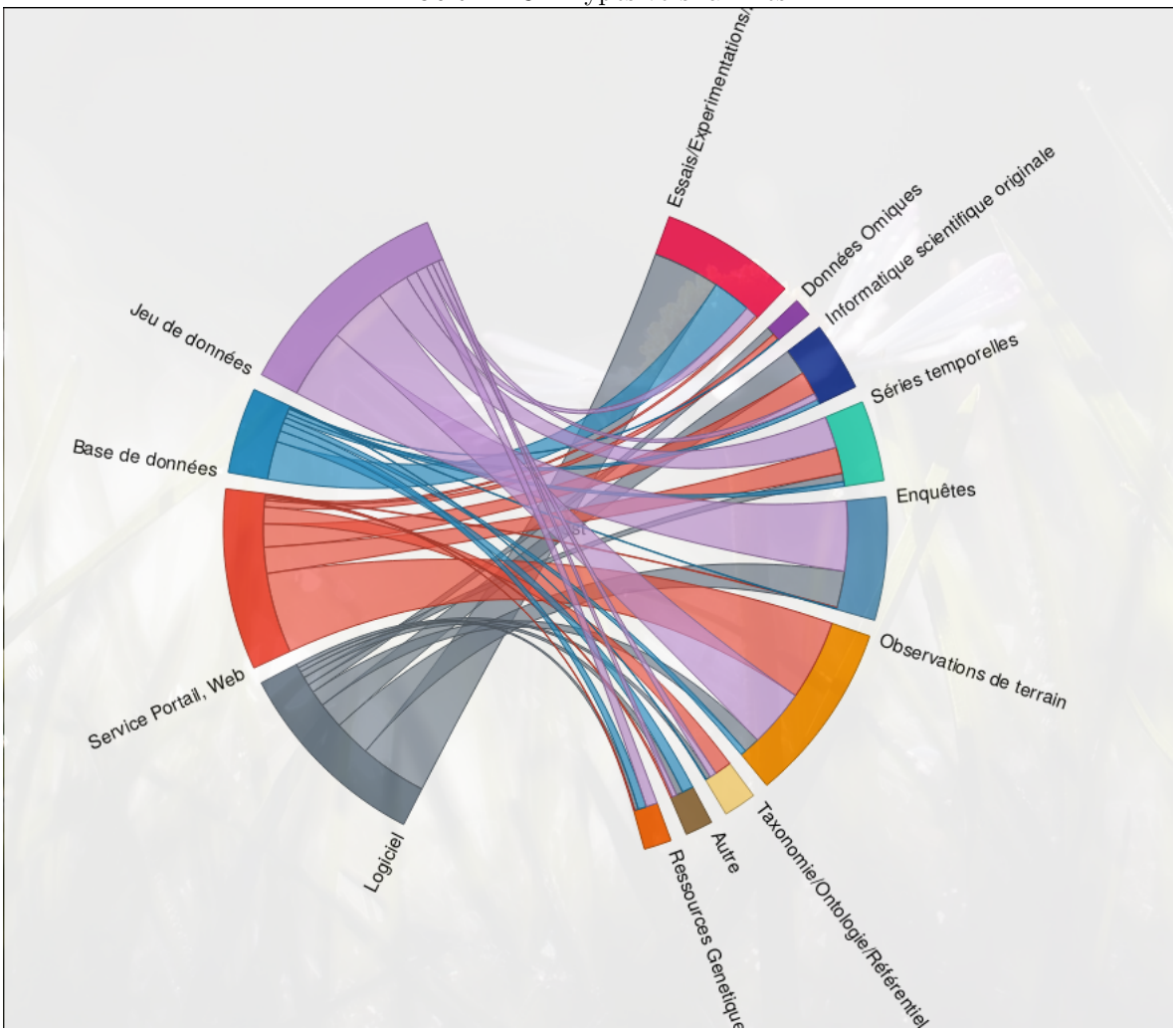
FIGURE 4.12 – Qui travaille avec qui ?



Relation Types vers Familles de données

Cette visualisation (voir figure 4.13) prend les quatre principaux types de données (jeu de données, base de données, service portail Web et logiciel) pour les relier à toutes les familles de données (séries temporelles, observations de terrain, enquêtes, etc.). Le choix de réduire les nombreux types de données à quelques uns est encore une fois motivé par la volonté de clarté et de lisibilité.

FIGURE 4.13 – Types vers familles



Ce diagramme nous donne l'information suivante : tant de familles de données sont de tel type de données. Et inversement il peut se lire comme : tant de types de données sont de telle famille de données. On peut donc lire ici que la majorité des enquêtes sont des jeux de données et qu'une majorité des logiciels concernent des essais/expérimentations/analyses laboratoire.

On notera la présence de deux zones vides servant de séparation. Ce sont deux groupes artificiels ajoutés à la matrice et rendus invisibles. D'ailleurs la matrice est un peu différente de celle présentée au début

de la section. Il s'agit toujours d'une matrice carrée mais est aussi symétrique. Sa taille est $n * n$ avec $n = nombreDeTypes + nombreDeFamilles + 2$.

Exemple de construction de chord diagram

Imaginons deux types et trois familles. Nous avons une matrice de la forme suivante :

$$\begin{pmatrix} 0 & 0 & 10 & 15 & 8 \\ 0 & 0 & 12 & 22 & 30 \\ 10 & 12 & 0 & 0 & 0 \\ 15 & 22 & 0 & 0 & 0 \\ 8 & 30 & 0 & 0 & 0 \end{pmatrix}$$

Pour créer la séparation on ajoute deux groupes d'une taille de 40 unités (par exemple), pointant l'un vers l'autre de manière égale :

$$\begin{pmatrix} 0 & 0 & 0 & 10 & 15 & 8 & 0 \\ 0 & 0 & 0 & 12 & 22 & 30 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 40 \\ 10 & 12 & 0 & 0 & 0 & 0 & 0 \\ 15 & 22 & 0 & 0 & 0 & 0 & 0 \\ 8 & 30 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Il ne reste plus qu'à rendre invisible les deux groupes ainsi que le ruban, et ceci à l'aide du CSS. Éventuellement il peut être nécessaire d'effectuer une rotation du diagramme pour s'adapter à l'ajout des deux groupes.

Relation Types vers Thématiques scientifiques

Le même diagramme a été réalisé pour montrer les relations entre les types et les thématiques. Comme il y a plus de thématiques que de familles, la représentation est plus dense et un peu moins visible. Voir la figure en annexe C.5.

Essais : relations Thématiques vers Thématiques et Filières vers Filières

Dans les annexes C.6 et C.7 on peut trouver des essais de chord diagram avec une densité importante. On remarquera que la densité est telle que cela devient illisible, ce pour quoi nous n'avons gardé que deux représentations de ce genre.

Chapitre 5

Conclusion

5.1 Bilan

L'objectif de mon stage était de réaliser l'intégration des données de l'inventaire du Cirad et ensuite de proposer des visualisations adaptées à cet ensemble. La première partie a occupé la majeure partie du temps, par une intense activité de scripting afin d'extraire et de normaliser les données. La deuxième partie a abouti à la création d'une application web permettant l'affichage et une visualisation avancée des données. Un formulaire y a été greffé en cours de route pour permettre l'ajout de données propres et fiables. Le contexte révèle que nous devons nous limiter à un nombre restreint de visualisations différentes, cela à cause de la nature des données et de leur propreté.

5.2 Perspectives

Cependant il reste quelques améliorations à réaliser, notamment sur ce dernier au niveau de la gestion de l'ajout des auteurs afin de le rendre plus fiable et plus efficace. Il manque encore quelques visualisations à mettre en place pour rendre la partie associée plus complète. Bien sûr il faut ajouter la gestion des utilisateurs qui est actuellement absente (elle n'était pas prioritaire). Tout cela sera poursuivi pendant le temps restant du stage.

Dans les prochains mois les chercheurs pourront donc bénéficier d'une nouvelle interface pour consulter ou ajouter des ressources dans l'inventaire ainsi qu'avoir une vue sur l'ensemble de celles-ci et faire des recherches et des évaluations sur ces corpus de métadonnées et données..

Annexe A

Configuration du serveur

A.1 Système

L'application web est hébergé sur une machine tournant avec la distribution linux CentOS 6.8. Python est utilisé dans sa version 3.4 et les packages sont installés avec pip. Le serveur est Apache et utilise mod_wsgi pour faire fonctionner Django.

A.2 Base de données

A.2.1 Droits et rôles

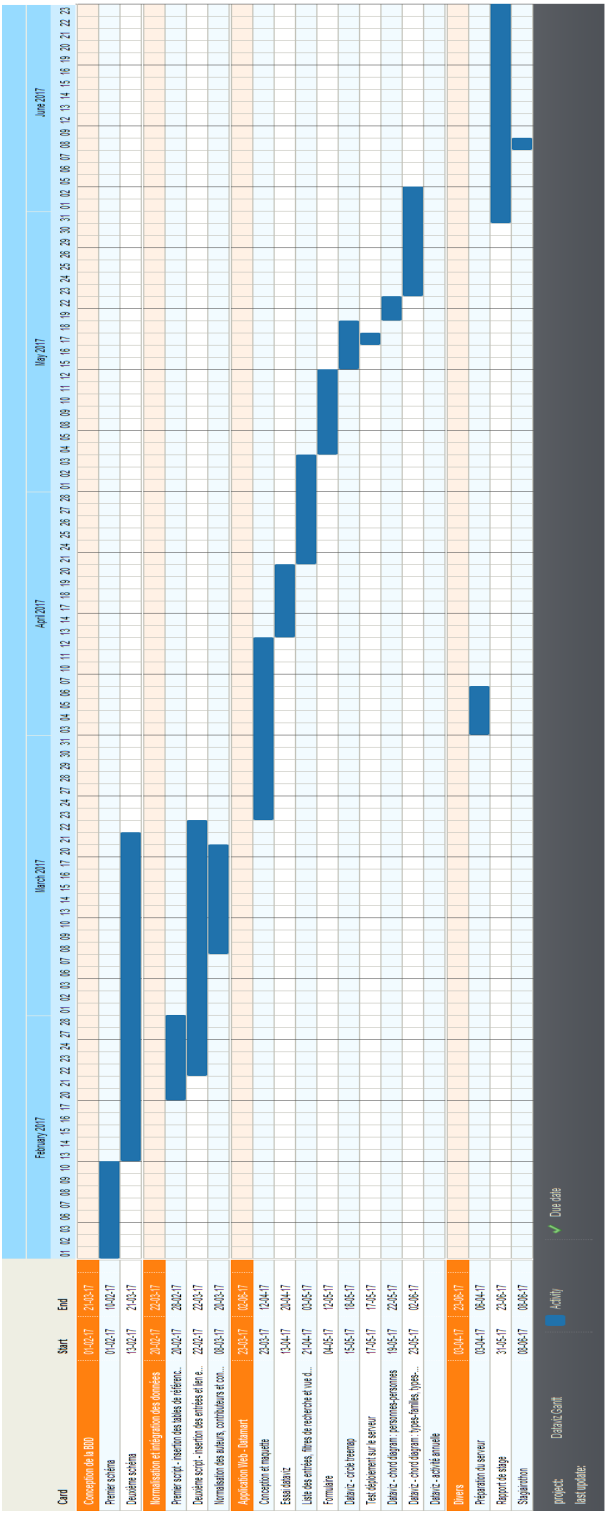
La base de données "inventaire" sera accessible par trois rôles :

- datapns_ a : rôle = administrateur et propriétaire de la BDD :
  - possède le droit de créer des bases de données ;
  - possède le droit de créer des rôles ;
- datapns_ c : rôle = create (CRUD) :
  - possède le droit USAGE sur tous les schémas ;
  - possède le droit USAGE sur toutes les séquences ;
  - possède les droits SELECT, INSERT, UPDATE et DELETE sur toutes les tables de tous les schémas ;
- datapns_ r : rôle = read :
  - possède le droit USAGE sur tous les schémas ;
  - possède le droit SELECT sur toutes les tables de tous les schémas ;

Annexe B

Planning prévisionnel

FIGURE B.1 – Schéma des actions de l'ensemble des scripts



Annexe C

Sources et images

C.8 Chord diagram : sources

```
function makeChordDiagram(url, id) {
  d3.json(url, function(error, imports) {
    if (error) throw error;

    var svg = d3.select(id),
      width = +svg.attr("width"),
      height = +svg.attr("height"),
      outerRadius = Math.min(width, height) * 0.5 - 150,
      innerRadius = outerRadius - 30;

    var formatValue = d3.formatPrefix(",.0", 1);

    var imports = imports.children;
    var listOfTypes = ["Logiciel", "Service Portail, Web", "Base de données", "Jeu de
    données"];

    var matrix = [];

    /*
     * Initialize matrix with zeros.
     * Size : number of Thematiques, etc. + number of Types
     * + 2 to add invisible arcs.
     */

    for (var i = 0; i < imports.length + 6; i++) {
      var new_line = [];

      for (var j = 0; j < imports.length + 6; j++) {
        new_line.push(0);
      }

      matrix.push(new_line);
    }

    matrix[imports.length][matrix.length - 1] = 200;
    matrix[matrix.length - 1][imports.length] = 200;

    for (var i = 0; i < imports.length; i++) {
```

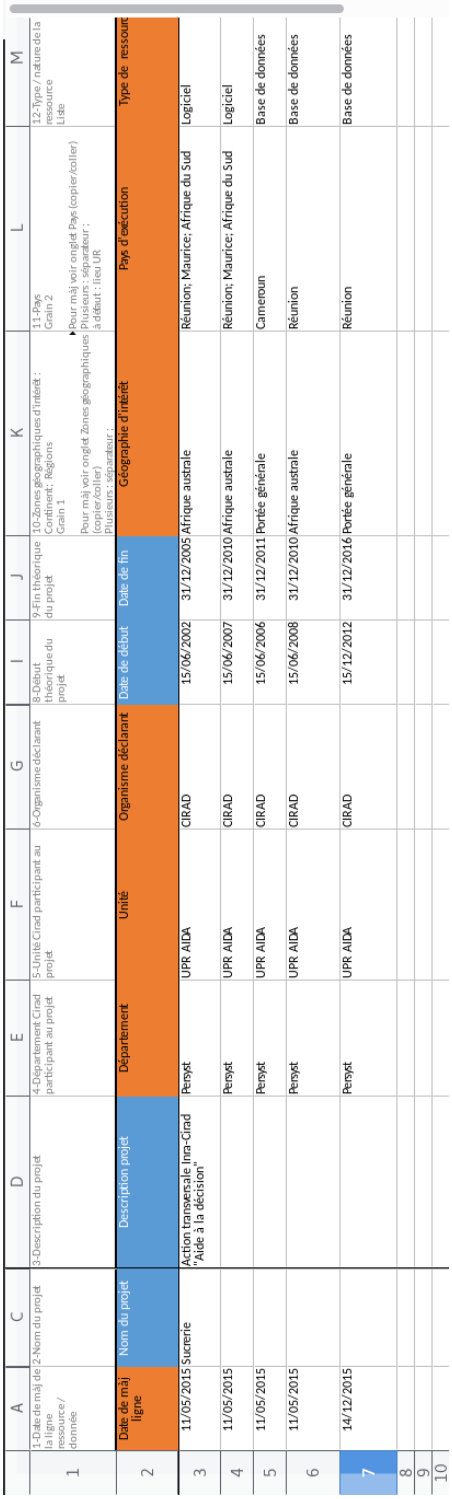
```
38        for (var j = imports.length + 1; j < imports.length + 5; j++) {
39          if (imports[i] && imports[i].children[j - imports.length - 1]) {
40            matrix[i][j] = imports[i].children[j - imports.length - 1].size;
41            matrix[j][i] = imports[i].children[j - imports.length - 1].size;
42          }
43        }
44      }
45
46      var offset = 0.35;
47
48      function startAngle(d) {
49        return d.startAngle + offset;
50      }
51
52      function endAngle(d) {
53        return d.endAngle + offset;
54      }
55
56      var chord = d3.chord()
57        .padAngle(0.05)
58        .sortSubgroups(d3.descending);
59
60      var arc = d3.arc()
61        .innerRadius(innerRadius)
62        .outerRadius(outerRadius)
63        .startAngle(startAngle)
64        .endAngle(endAngle);
65
66      var ribbon = d3.ribbon()
67        .radius(innerRadius)
68        .startAngle(startAngle)
69        .endAngle(endAngle);
70
71      var color = d3.scaleOrdinal()
72        .domain(d3.range(20))
73        .range(["#F62459", "#8E44AD", "#1F3A93", "#36D7B7", "#5C97BF",
74          "#F89406", "#FFDD89", "#957244", "#F9690E", "#FDE3A7",
75          "#6C7A89", "#EF4836", "#1E8BC3", "#BE90D4", "#FDE3A7",
76          "#E26A6A", "#81CFE0", "#19B5FE", "#03C9A9", "#F2784B"
77        ]);
78
79      var g = svg.append("g")
80        .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")")
81        .datum(chord(matrix));
82
83      var group = g.append("g")
84        .attr("class", "groups")
85        .selectAll("g")
86        .data(function(chords) {
87          return chords.groups;
88        })
89        .enter().append("g");
90
91      group.append("path")
92        .style("fill", function(d) {
93          return color(d.index);
94        })
95        .style("stroke", function(d) {
96          return d3.rgb(color(d.index)).darker();
97        })
98        .style("opacity", function(d) {
99          if (d.index == imports.length || d.index == imports.length + 5) {
```

```
100             return 0;
101           } else {
102             return 1;
103           }
104         })
105         .attr("d", arc);
106
107       var groupTick = group.selectAll(".group-tick")
108         .data(function(d) {
109           return groupTicks(d, 30);
110         })
111         .enter().append("g")
112         .attr("class", "group-tick")
113         .attr("transform", function(d) {
114           return "rotate(" + ((d.angle) * 180 / Math.PI - 90) + ") translate(" +
      outerRadius + ",0)";
115         });
116
117       groupTick
118         .filter(function(d) {
119           return d.value % 200 === 0;
120         })
121         .append("text")
122         .attr("x", 8)
123         .attr("dy", ".35em")
124         .attr("transform", function(d) {
125           return d.angle > Math.PI ? "rotate(180) translate(-16)" : null;
126         })
127         .style("text-anchor", function(d) {
128           return d.angle > Math.PI ? "end" : null;
129         })
130         .text(function(d, i) {
131           return getLabel(d.label);
132         });
133
134       g.append("text")
135         .each(function(d) {
136           d.angle = (d.startAngle + d.endAngle) / 2;
137         })
138         .attr("dy", ".35em")
139         .attr("transform", function(d) {
140           return "rotate(" + (d.angle * 180 / Math.PI - 90) + ")" +
141             "translate(" + (innerRadius + 26) + ")" +
142             (d.angle > Math.PI ? "rotate(180)" : "");
143         })
144         .style("text-anchor", function(d) {
145           return d.angle > Math.PI ? "end" : null;
146         })
147         .text(function(d) {
148           return "test";
149         });
150
151
152       g.append("g")
153         .attr("class", "ribbons")
154         .selectAll("path")
155         .data(function(chords) {
156           return chords;
157         })
158         .enter().append("path")
159         .attr("d", ribbon)
160         .style("fill", function(d) {
```

```
        return color(d.target.index);
      })
      .style("stroke", function(d) {
        return d3.rgb(color(d.target.index)).darker();
      })
      .style("opacity", function(d) {
        if (d.target.index == imports.length || d.target.index == imports.length + 5) {
          return 0;
        } else {
          return 1;
        }
      });

    function getLabel(index) {
      if (imports[index]) {
        return imports[index].name;
      } else if (index == imports.length || index == imports.length + 5) {
        return "";
      } else {
        return listOfTypes[index - imports.length - 1];
      }
    }

    // Returns an array of tick angles and values for a given group and step.
    function groupTicks(d, step) {
      var k = (d.endAngle - d.startAngle) / d.value;
      return d3.range(0, d.value, step).map(function(value) {
        return {
          value: value,
          angle: value * k + d.startAngle + offset,
          label: d.index
        };
      });
    }

    function fade(opacity) {
      return function(g, i) {
        svg.selectAll("path")
          .filter(function(d) {
            return d.index != i && d.subindex != i; // && i != imports.length;
          })
          .transition()
          .style("opacity", opacity);
      };
    }

  });
}

makeChordDiagram("/dataviz/json/sometypesby/Famille/", "#svg-types-familles");
makeChordDiagram("/dataviz/json/sometypesby/Thematique/", "#svg-types-thematiques");
```

C.1 Exemple de document Excel

FIGURE C.1 – Exemple de document Excel

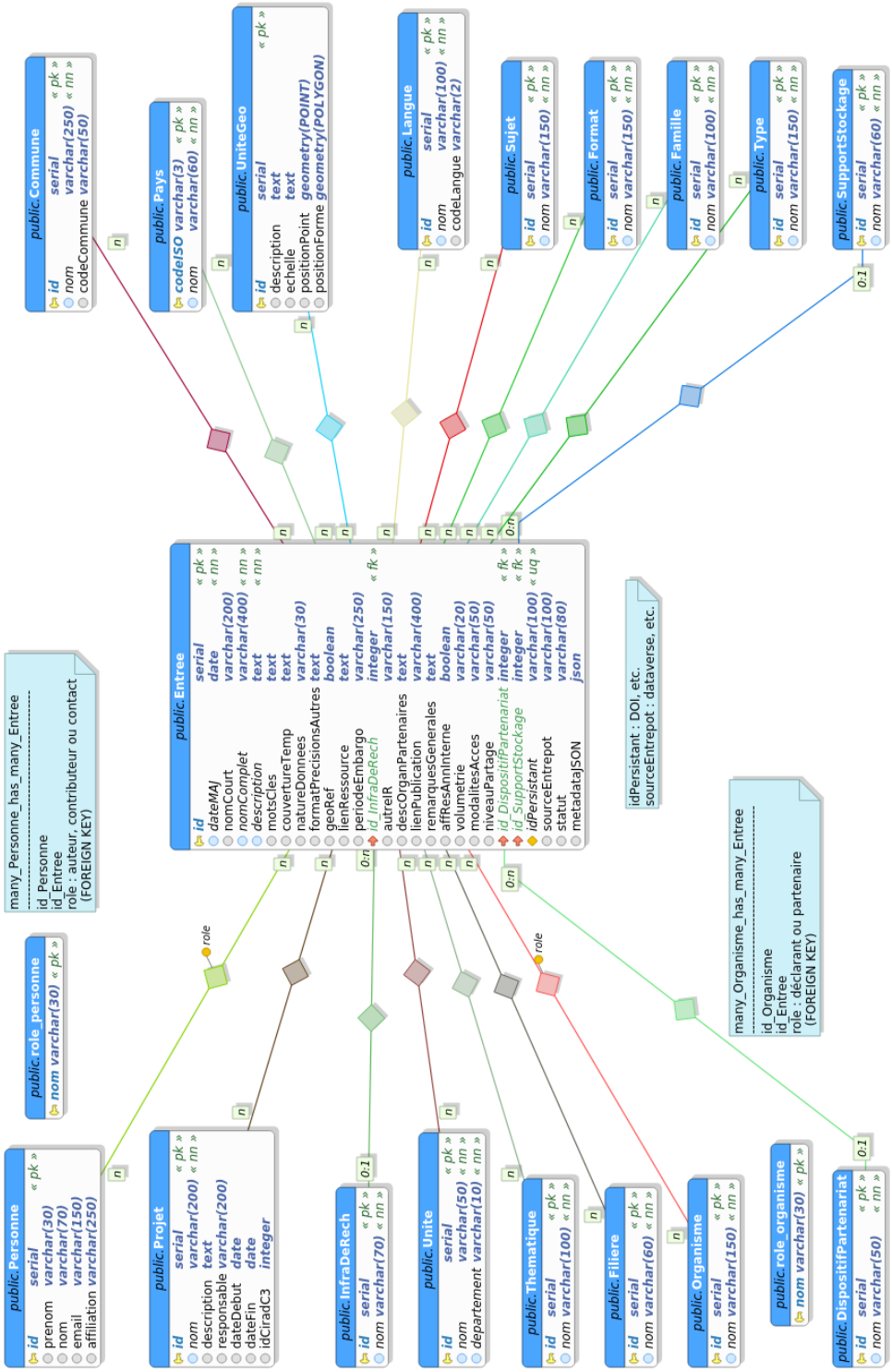


	A	C	D	E	F	G	I	J	K	L	M
1	1-Date de màj de la ligne ressource / donnée	2-Nom du projet	3-Description du projet	4-Département Cirad participant au projet	5-Unité Cirad participant au projet	6-Organisme déclarant	8-Début théorique du projet	9-Fin théorique du projet	10-Zones géographiques d'intérêt : Continent; Régions Grain 1 Pour màj voir onglet Zones géographiques (copier/coller) Plusieurs : séparateur ;	11-Pays Grain 2 Pour màj voir onglet Pays (copier/coller) Plusieurs : séparateur ; à défaut : lieu UR	12-Type / nature de la ressource Liste
2	Date de màj ligne	Nom du projet	Description projet	Département	Unité	Organisme déclarant	Date de début	Date de fin	Géographie d'intérêt	Pays d'exécution	Type de ressour
3	11/05/2015	Sucrerie	Action transversale Inra-Cirad "Aide à la décision"	Persyst	UPR AIDA	CIRAD	15/06/2002	31/12/2005	Afrique australe	Réunion; Maurice; Afrique du Sud	Logiciel
4	11/05/2015			Persyst	UPR AIDA	CIRAD	15/06/2007	31/12/2010	Afrique australe	Réunion; Maurice; Afrique du Sud	Logiciel
5	11/05/2015			Persyst	UPR AIDA	CIRAD	15/06/2006	31/12/2011	Portée générale	Cameroun	Base de données
6	11/05/2015			Persyst	UPR AIDA	CIRAD	15/06/2008	31/12/2010	Afrique australe	Réunion	Base de données
7	14/12/2015			Persyst	UPR AIDA	CIRAD	15/12/2012	31/12/2016	Portée générale	Réunion	Base de données
8											
9											
10											

FIGURE C.2 – Ancienne interface Web

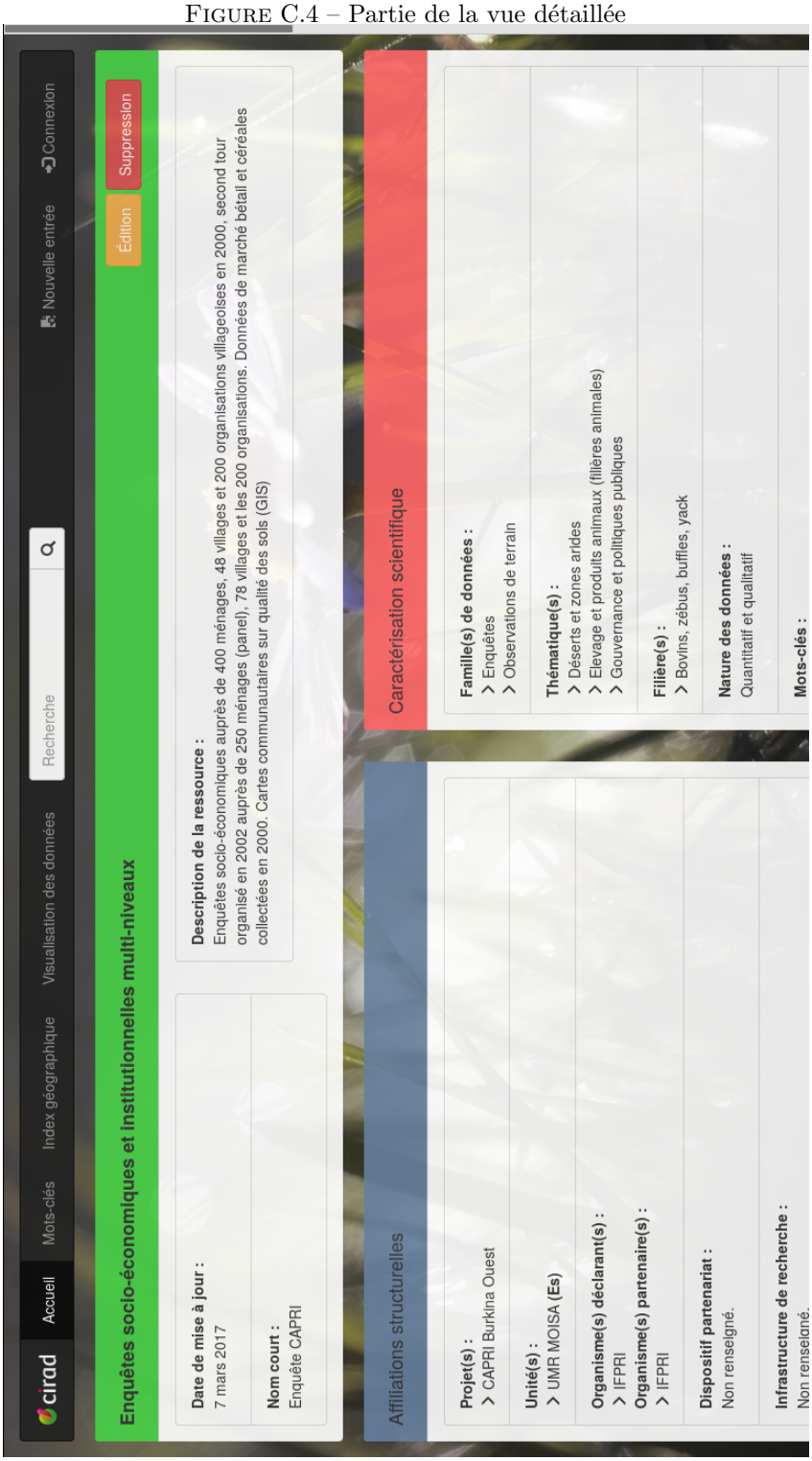
C.3 Nouveau schéma de la base de données

FIGURE C.3 – Inventaire



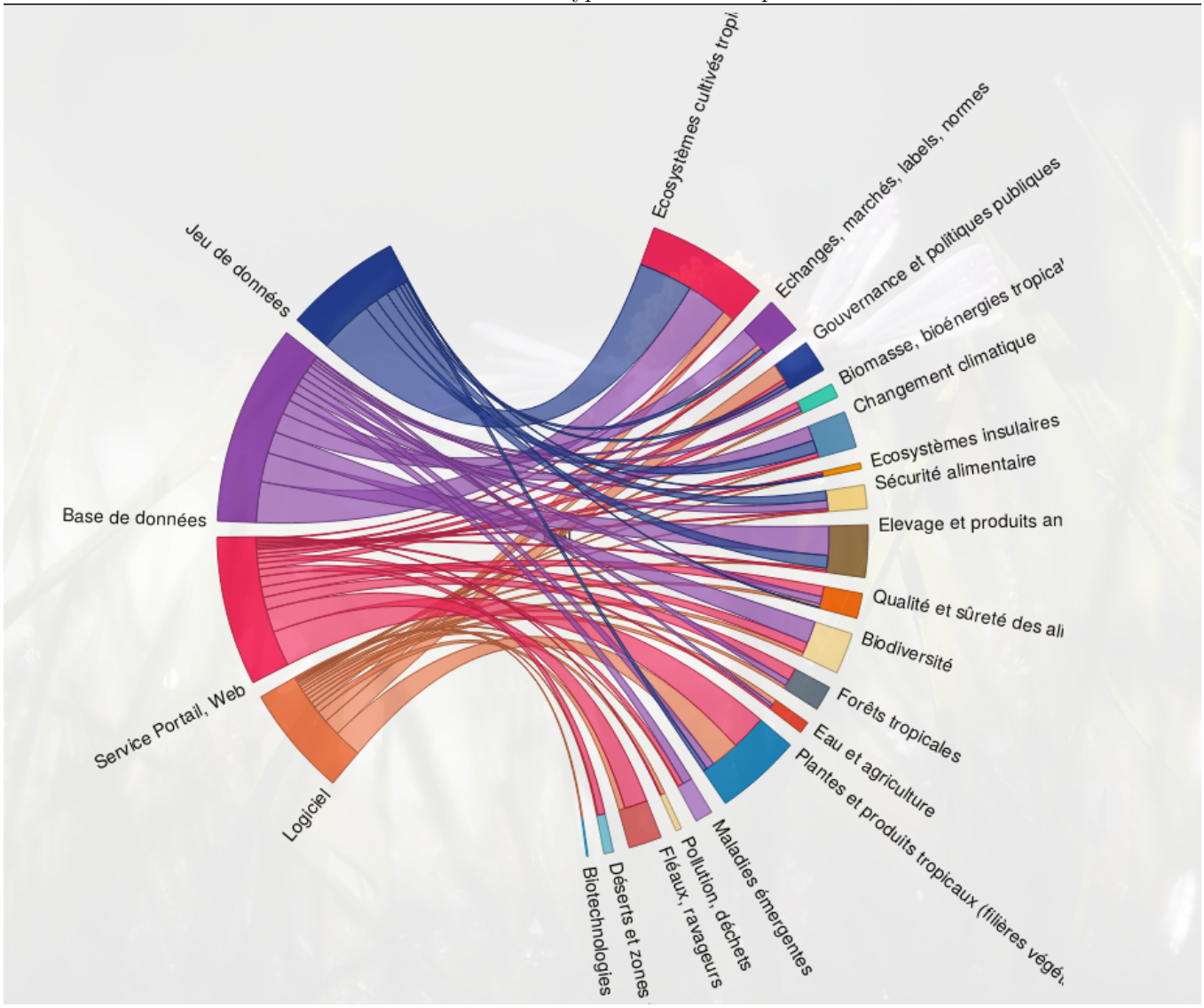
C.4 Vue détaillée

FIGURE C.4 – Partie de la vue détaillée



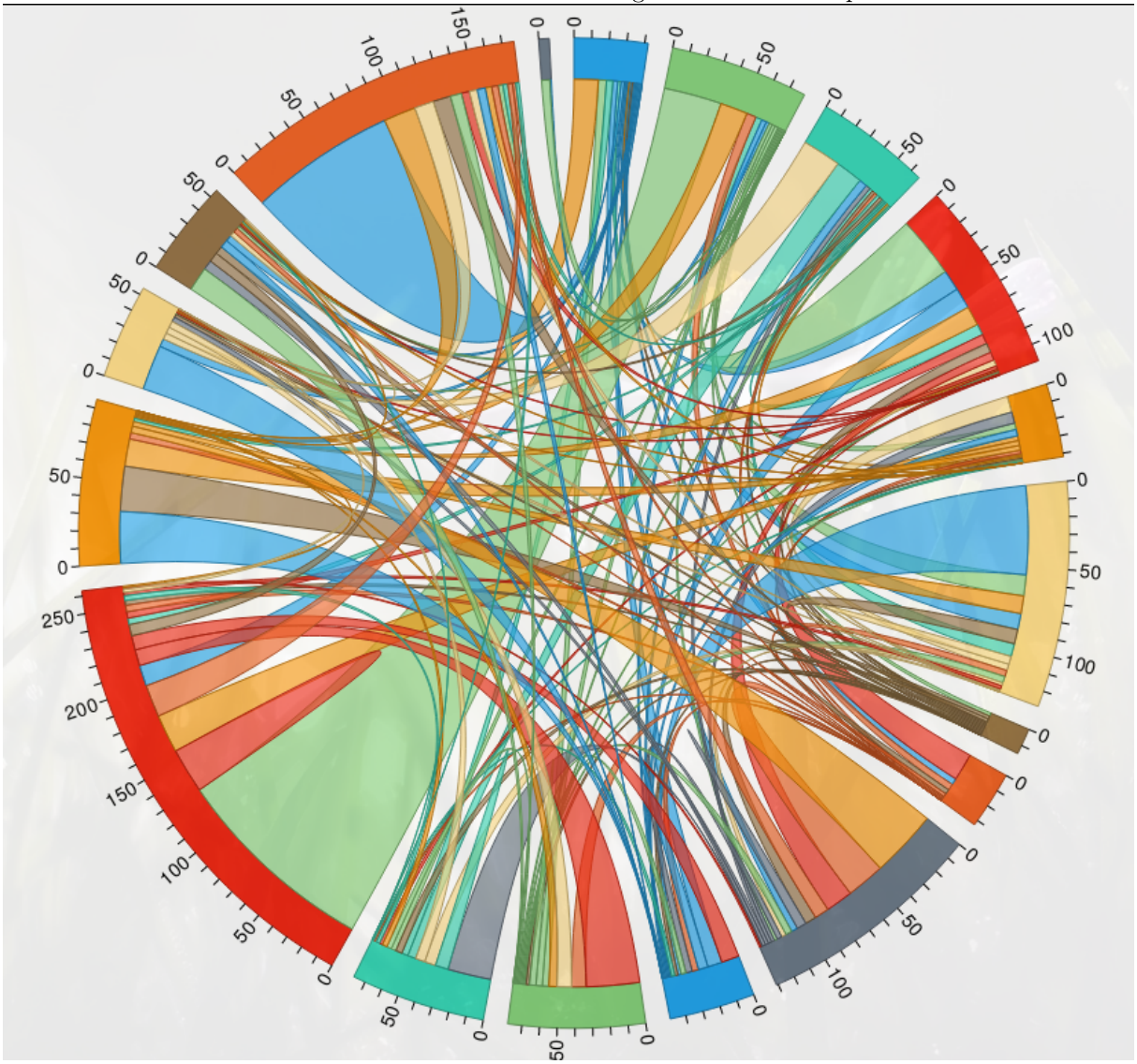
C.5 Chord diagram : Types-Thématiques

FIGURE C.5 – Types vers thématiques



C.6 Chord diagram : Thématiques-Thématiques

FIGURE C.6 – Essai d'un chord diagram inter-thématiques



C.7 Chord diagram : Filières-Filières

FIGURE C.7 – Essai d'un chord diagram inter-filières